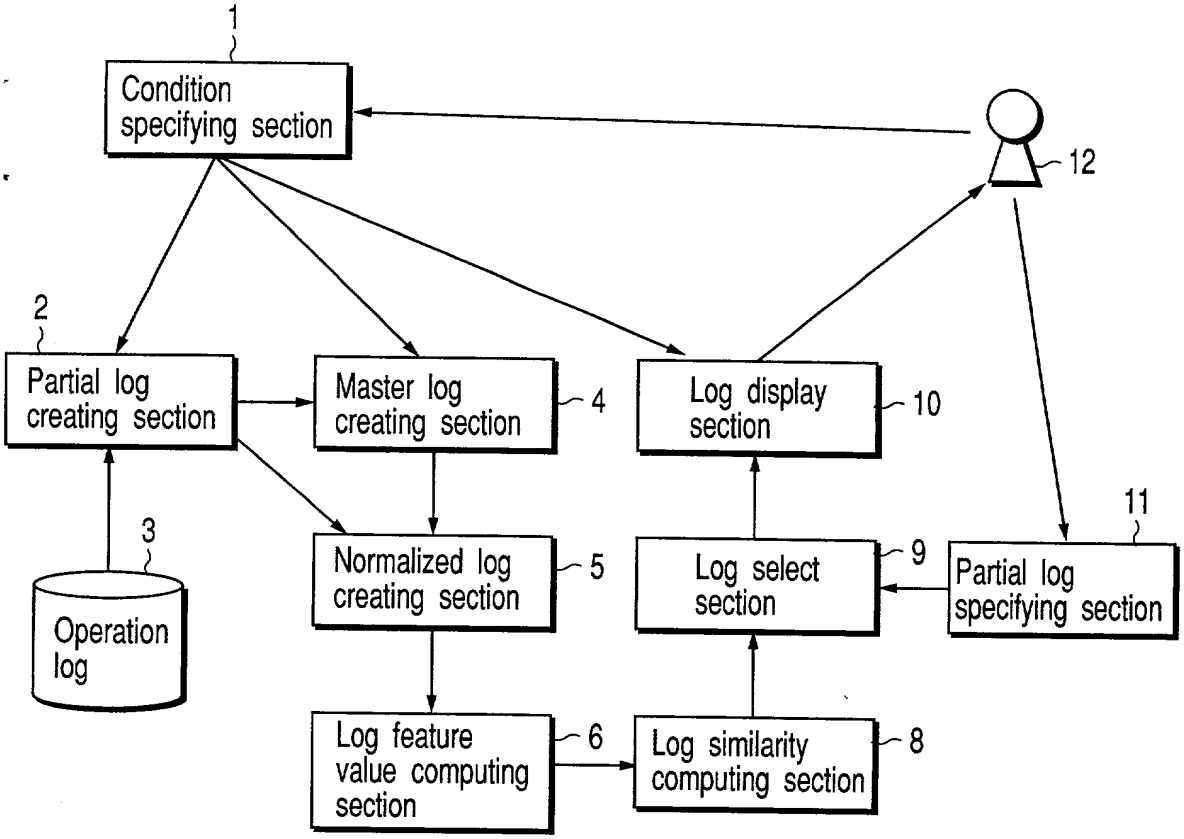

1
Condition specifying section
12
2
Partial log creating section
Master log creating section
4
Log display section
10
11
3
Operation log
Normalized log creating section
5
Log select section
9
Partial log specifying section
Log feature value computing section
6
Log similarity computing section
8

FIG. 1

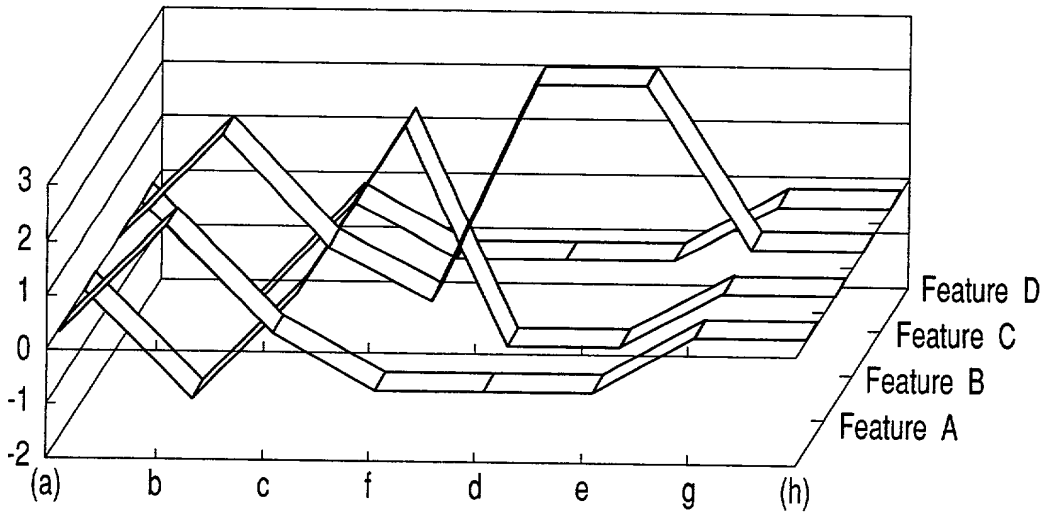

3
2
1
0
-1
-2
(a)
b
c
f
d
e
g
(h)
Feature D
Feature C
Feature B
Feature A

FIG. 3

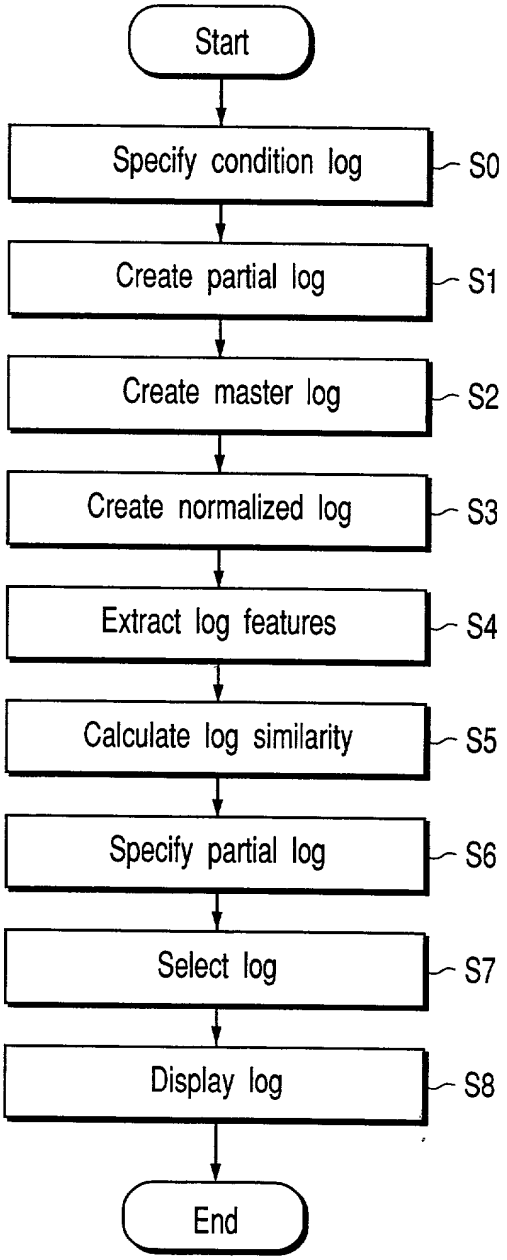

Start
Specify condition log
S0
Create partial log
S1
Create master log
S2
Create normalized log
S3
Extract log features
S4
Calculate log similarity
S5
Specify partial log
S6
Select log
S7
Display log
S8
End

FIG. 2

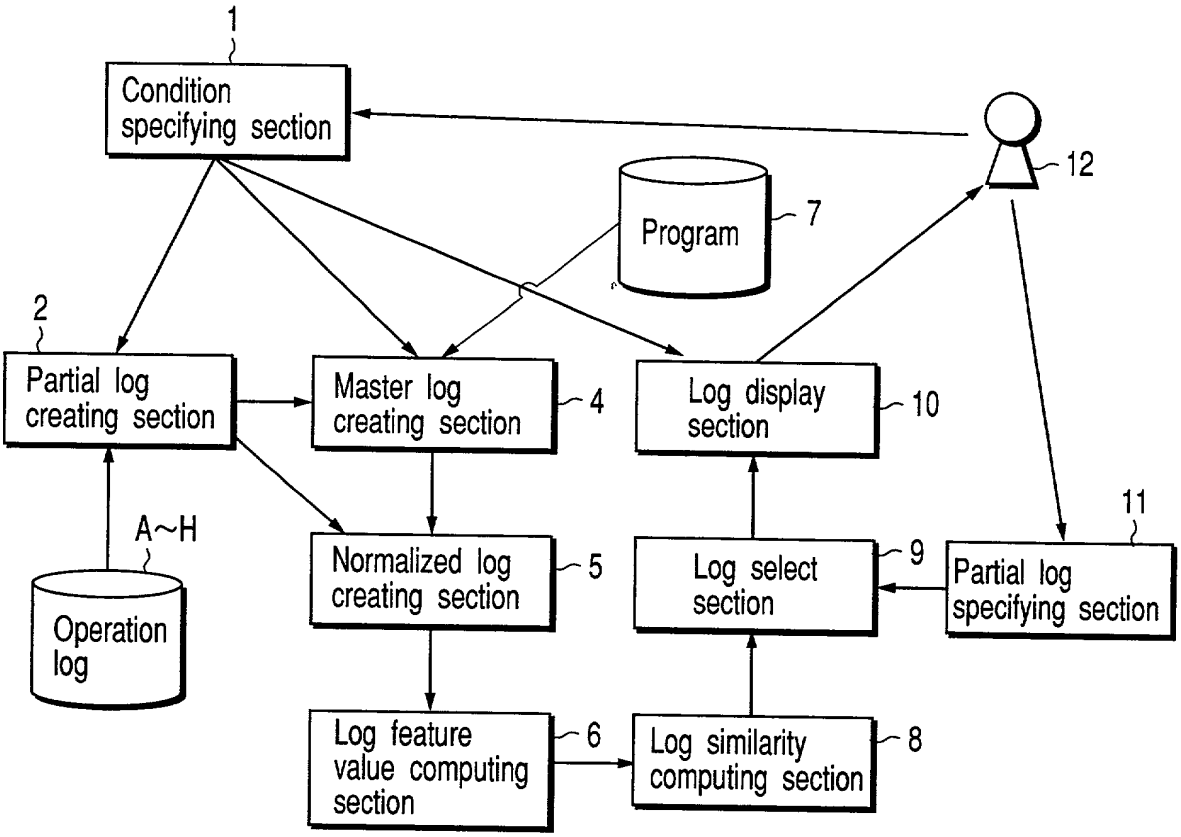

1
Condition specifying section
12
Program
7
2
Partial log creating section
Master log creating section
4
Log display section
10
11
A~H
Normalized log creating section
5
Log select section
9
Partial log specifying section
Operation log
Log feature value computing section
6
Log similarity computing section
8

FIG. 4

```
#include<stdio.h>
#define MAX_STR_NUM 10

// declaration of functions
void PrintError();
void reverseString(char* string);
void makeValueString(int value,char* str);
// main functions
int main(int argc,char** argv){
    char string[MAX_STR_NUM];
    // convert the number of arguments into character strings
    in ternary representation (the order of characters is in reverse)
    makeValueString(argc,string);
    // reverse the order of characters
    reverseString(string);
    // display the result
    printf(" %d to %s\n",argc,string);
  }

// definition of each function
void makeValueString(int value,char* str);
  {
    // recursion end condition for recursive function
    if(value<=0)
      {
        str[0]='\0';
        return;
      }
    makeValueString( value/3,str+1);
    switch(value%3)
      {
      case 0:
        str[0]='\0'; // mistaken for '0'
        break;
      case 1:
        str[0]='1';
        break;
      case 2:
        str[0]='2';
        break;
      default:
        break;
```

FIG. 5A

```
    }
  }
void PrintError()
  {
    printf("error\n");
  }
void reverseString(char* string)
  {
    char tmp_char
    int n;
    int i;
    n=strlen(string);
    if(n==0)
      {
        PrintError(); // error process
      }
    else
      {
       // reverse the order of the character strings
       for (i=0;i<(n/2);i++)
         {
           tmp_char=string[i];
           string[i]=string[n-1-i];
           string[n-1-i]=tmp_char;
         }
      }
  }
```

FIG. 5B

```
main(12,0x10000)
  {
    makeValueString(12,0x20000)
      {
        if(value<=0)
        makeValueString(4,0x20001)
          {
            if(value<=0)
            makeValueString(1,0x20002)
              {
                if(value<=0)
                makeValueString(0,0x20003)
                  {
                    if(value<=0)
                      {
                      }
                  }
                switch(value%3)
                  {
                  case 1:
                  }
              }
            switch(value%3)
              {
              case 1:
              }
          }
        switch(value%3)
          {
          case 0:
          }
      }
    reverseString(0x20000)
      {
        n=strlen(0x20000);
        if(n==0)
          {
            PrintError()
              {
                printf(" error\n");
              }
          }
      }
    printf(" %d to %s\n",12,0x20000);
  }
```

FIG. 6

```
main(13,0x10000)
  {
    makeValueString(13,0x20000)
      {
        if(value<=0)
        makeValueString(4,0x20001)
          {
            if(value<=0)
            makeValueString(1,0x20002)
              {
                if(value<=0)
                makeValueString(0,0x20003)
                  {
                    if(value<=0)
                      {
                      }
                  }
                switch(value%3)
                  {
                  case 1:
                  }
              }
            switch(value%3)
              {
              case 1:
              }
          }
        switch(value%3)
          {
          case 1:
          }
      }
    reverseString(0x20000)
      {
        n=strlen(0x20000);
        if(n==0)
        else
          {
            for(i=0,i<(n/2);i++)
              {
              }
          }
      }
    printf(" %d to %s\n",13,0x20000);
  }
```

FIG. 7

```
int main(int argc,char** argv)
  {
    makeValueString(int value,char* str)
      {
        if(value<=0)
          {
          }
        makeValueString(int value,char* str)
          {
            if(value<=0)
              {
              }
            makeValueString(int value,char* str)
              {
                if(value<=0)
                  {
                  }
                makeValueString(int value,char* str)
                  {
                    if(value<=0)
                      {
                      }
                    makeValueString(value/3,str+1);
                    switch(value%3)
                      {
                      case 0:
                      case 1:
                      case 2:
                      }
                  }
                switch(value%3)
                  {
                  case 0:
                  case 1:
                  case 2:
                  }
              }
            switch(value%3)
              {
              case 0:
              case 1:
              case 2:
              }
          }
```

FIG. 8A

```
        switch(value%3)
            {
            case 0:
            case 1:
            case 2:
            }
    }
reverseString(char* string)
    {
      n=strlen(string);
      if(n==0)
          {
            PrintError()
              {
                printf(" error\n");
              }
          }
       else
          {
            for(i=0,i<(n/2);i++)
              {
              }
            for(i=0,i<(n/2);i++)
              {
              }
            for(i=0,i<(n/2);i++)
              {
              }
          }
    }
  printf(" %d to %s\n",argc,string);
}
```

FIG. 8B

```
1 main()
1 {
1   makeValueString()
1     {
1       if(value<=0)
0         {
0         }
1       makeValueString()
1         {
1           if(value<=0)
0             {
0             }
1           makeValueString()
1             {
1               if(value<=0)
0                 {
0                 }
1               makeValueString()
1                 {
1                   if(value<=0)
1                     {
1                     }
0                   makeValueString();
1                   switch(value%3)
1                     {
0                     case 0:
1                     case 1:
1                     case 2:
1                     }
1                 }
1                 switch(value%3)
1                   {
0                   case 0:
1                   case 1:
0                   case 2:
1                   }
1               }
1               switch(value%3)
1                 {
1                 case 0:
0                 case 1:
0                 case 2:
1                 }
```

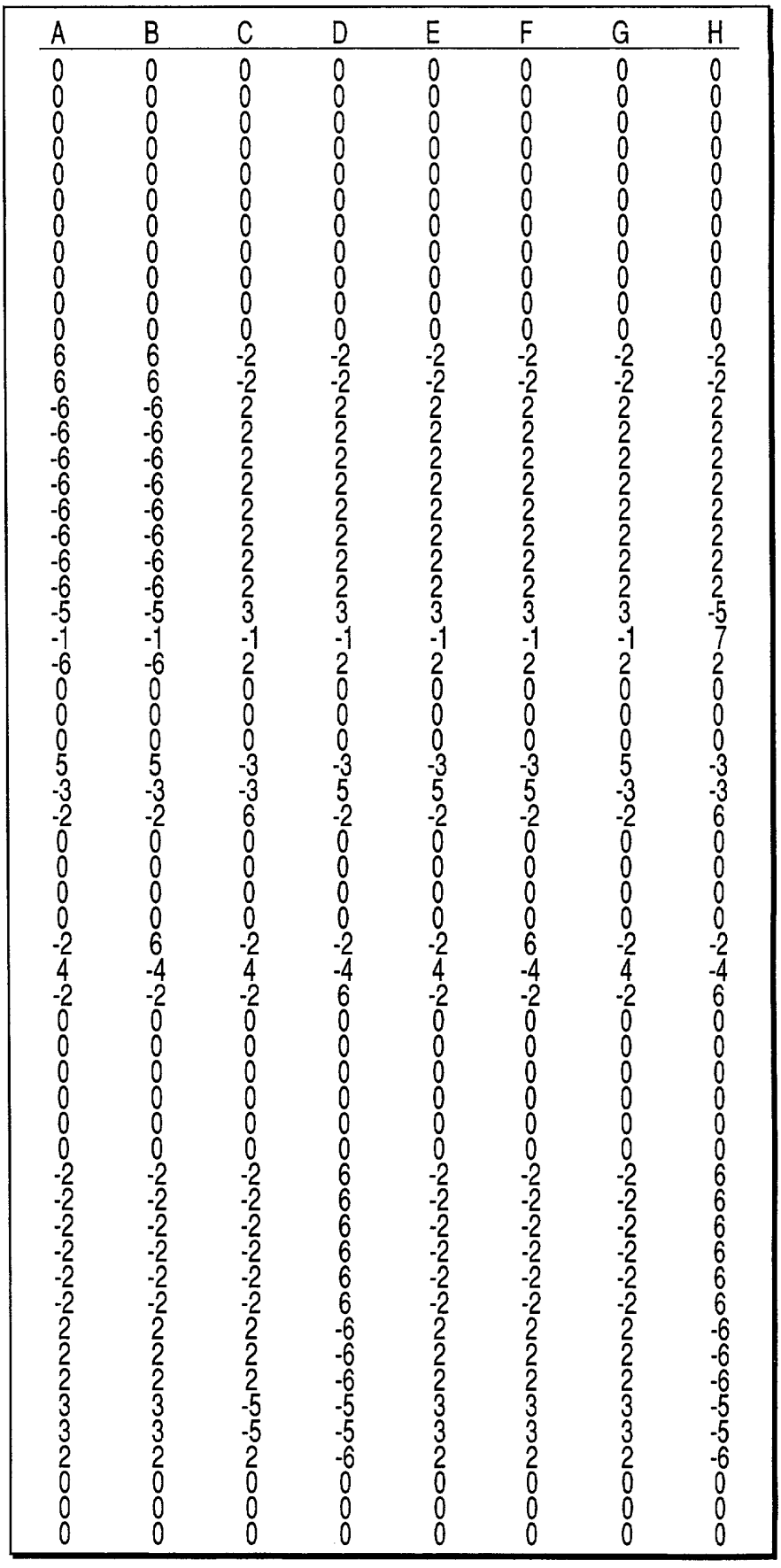
FIG. 9A

```
1              }
1            switch(value%3)
1              {
1              case 0:
0              case 1:
0              case 2:
1              }
1          }
1        reverseString()
1          {
1            n=strlen();
1            if(n==0)
1              {
1                PrintError()
1                  {
1                    printf();
1                  }
1              }
0            else
0              {
0                for(i=0,i<(n/2);i++)
0                  {
0                  }
0                for(i=0,i<(n/2);i++)
0                  {
0                  }
0                for(i=0,i<(n/2);i++)
0                  {
0                  }
0              }
0          }
1          printf();
1        }
```

FIG. 9B

A	B	C	D	E	F	G	H
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	0
0	0	0	0	0	0	0	1
0	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	0	0	0	0	1	0
0	0	0	1	1	1	0	0
0	0	1	0	0	0	0	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
0	1	0	0	0	1	0	0
1	0	1	0	1	0	1	0
0	0	0	1	0	0	0	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1
1	1	1	0	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	0	1	1	1	0
1	1	0	0	1	1	1	0
1	1	0	0	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

FIG. 10



A	B	C	D	E	F	G	H
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
6	6	-2	-2	-2	-2	-2	-2
6	6	-2	-2	-2	-2	-2	-2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-6	-6	2	2	2	2	2	2
-5	-5	3	3	3	3	3	-5
-1	-1	-1	-1	-1	-1	-1	7
-6	-6	2	2	2	2	2	2
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
5	5	-3	-3	-3	-3	5	-3
-3	-3	-3	5	5	5	-3	-3
-2	-2	6	-2	-2	-2	-2	6
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-2	6	-2	-2	-2	6	-2	-2
4	-4	4	-4	4	-4	4	-4
-2	-2	-2	6	-2	-2	-2	6
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-2	-2	-2	6	-2	-2	-2	6
-2	-2	-2	6	-2	-2	-2	6
-2	-2	-2	6	-2	-2	-2	6
-2	-2	-2	6	-2	-2	-2	6
-2	-2	-2	6	-2	-2	-2	6
-2	-2	-2	6	-2	-2	-2	6
2	2	2	-6	2	2	2	-6
2	2	2	-6	2	2	2	-6
2	2	2	-6	2	2	2	-6
3	3	-5	-5	3	3	3	-5
3	3	-5	-5	3	3	3	-5
2	2	2	-6	2	2	2	-6
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

FIG. 11

	A	B	C	D	E	F	G	H
A		494	-130	-346	-90	-138	-26	-306
B			-178	-330	-138	-58	-74	-290
C				-58	70	22	70	-18
D					-82	-66	-146	470
E						126	110	-170
F							62	-154
G								-170
H								

FIG. 12